A Fact–Oriented Data Distribution System

Samuel C. Chamberlain

US Army Ballistic Research Laboratory,
Aberdeen Proving Grounds, MD 21005–5066

ABSTRACT

Digital data is increasingly being used as the means by which information is exchanged on the battlefield. Currently, digital information is exchanged using preformatted messages, graphics symbols, and a variety of mixed media forms that use more communications bandwidth than is practically available at the fighting echelons of the Army. However, by using the *data abstractions of military concepts* as the basic units of exchange, bandwidth requirements are reduced and information is represented in a more expressible manner. This requires equal consideration of both computer and military science.

The Ballistic Research Laboratory (BRL) is building an experimental information distribution system based on such data abstractions, named facts, that includes a RAM resident storage facility (factbase), a connectionless communications protocol (Fact Exchange Protocol), and explicit information to describe the data abstractions and strict rules that control the exchange of facts with other tactical nodes.

INTRODUCTION

The Ballistic Research Laboratory's (BRL) System Engineering and Concepts Analysis Division (SECAD) is developing a fire support application to demonstrate these information distribution concepts for the Army Distributed Command & Control Project (ADCCP). The tactical theme of the application will revolve around a dynamic fighting force at brigade and battalion level. The focus will be on the operations at five key artillery nodes (the maneuver brigade fire support element, a direct support field artillery battalion operations element, and three maneuver battalion fire support elements) who are responsible for two key fire support functions: *fire support control & coordination* (FSCC) and *field artillery tactical operations* (FA TAC OPS). These players will interact in response to several tactical vignettes that could occur as a result of a critical situation (e.g., a surprise attack) thus forcing them to change plans on–the–run, reconfigure forces and the command chain, conduct an operation in a secure and silent mode, and dynamically respond to battlefield losses.

A major goal is to develop tactical computer science technology that can support "fighting level" commanders and soldiers who must contend with a highly dynamic, unpredictable, and hostile tactical environment. Current tactical command and control systems support digital data exchange via preformatted "messages" (character strings) or graphics symbols that must be interpreted by a human operator. These systems fall short in two key respects: first, the information is not in a form that computers can manipulate in a sophisticated manner (i.e., "understand"), and second, they use inefficient protocols to exchange the information that require more bandwidth than would be practically available at the lower echelon fighting units that must use VHF–FM and HF–AM radios; this inefficiency also produces large electromagnetic signatures as a result of the time required to broadcast voluminous information.

The underlying purpose of any information distribution system is to support the exchange of ideas or concepts. At the superficial level this may simply be a unit's location, but underlying such a simple concept are the reasons for needing to know the unit's location. It is this type of "meta–information" that can be used to define the structure of tactical information and regulate its flow between the various nodes of the system. If a terse, efficient means of

communicating between the nodes of the system is to be developed, it is important to incorporate military science to define the data abstractions of the basic concepts or primitives. By using these primitives as the basic unit of exchange between command and control nodes, a more flexible and survivable system results (a sharp contrast to the technique of trying to parse incoming character strings into data abstractions). This requires that a significant effort be expended "up–front" as combat developers and computer scientists work together to build the data abstractions of the military concepts. Using this scheme, one of the task of high–level user application programs becomes converting data abstractions into a form suitable for user assimilation and manipulation.

The BRL "fact oriented data distribution system" incorporates several new concepts in an effort to explore techniques that provide more flexibility and survivability to the information distribution function in command and control systems while also providing the information in a form suitable for manipulation by sophisticated computer programs. Flexibility is enhanced via a freeform *distributed factbase* and associated *fire support control capability profiles*. Survivability is enhanced through minimizing electronic emanations by taking advantage of "overheard" information, providing a "radio silence" (Emission Control, or EMCON) mode of operation, and using multicast transmissions when possible. These features are being implemented in a *fact exchange protocol* that will exploit these and other features to create an "information stingy" protocol. The underlying objective is to build a *computationally* intensive rather than *communications* intensive protocol; i.e., "don't send information if it can be computed". **Figure 1** shows the basic structure of the fact distribution system. The distributed factbase (DFB) is composed of four conceptual modules: the factbase that stores the information, the Fact Exchange Protocol (FEP) used to exchange facts between factbases, the Package Protocol used to connect application programs with a factbase, and the Security Control Module (SCM) that controls DFB access.

FACTS and FACTBASES

All information is stored in a RAM resident, free–form, distributed factbase (DFB) as a collection of many interconnected *facts*. A fact is an instantiation of one of several predefined *fact types* that have a form tailored for describing the particular item, activity, or event that it represents; in other words, the fact types are the data abstractions of the military concepts and the facts are the actual information. As each fact is *stated* it is assigned a unique fact identification number, or *fact id*. The fact id is eight bytes wide: the first four bytes are the Arpanet host address of the host on which the fact was stated and the second four bytes are determined by the host itself (e.g., increasing integers). A fact consists of a header and one or more *fact items*. Like any data structure, each fact item ismore defined as a particular data type; in this implementation there are five types: integers, floating point numbers, character strings, references (fact ids of other facts), and lists of these five data types.

A fact type can be put into one of three categories: dynamic facts, reference material, and meta–facts. *Dynamic facts* are created (stated) at the hosts by its users or application programs and describe a dynamic event or activity. Since they are stated at the host they have a fact id whose first four bytes are the host address of that host, and this identifies where the fact originated when it is disseminated throughout the system. *Reference material* are facts with fixed facts ids that are common to every host and describe static "reference" information that typically would be found in field and technical manuals. This includes information about the Tables of Organization and Equipment (TO&E), vehicles, equipment, ammunition, etc. These facts are loaded into each factbase when it is initialized and never have to be updated. (Actually, this information is stored in a core image of the factbase code so that it is immediately available when the factbase is executed.) Since all the hosts have common fact ids for this information the fact ids can be passed when referring to the entity thus significantly reducing the amount of data that must be transmitted. Also included as preloaded reference material are the actual units (specific military organizations such as the "1–51

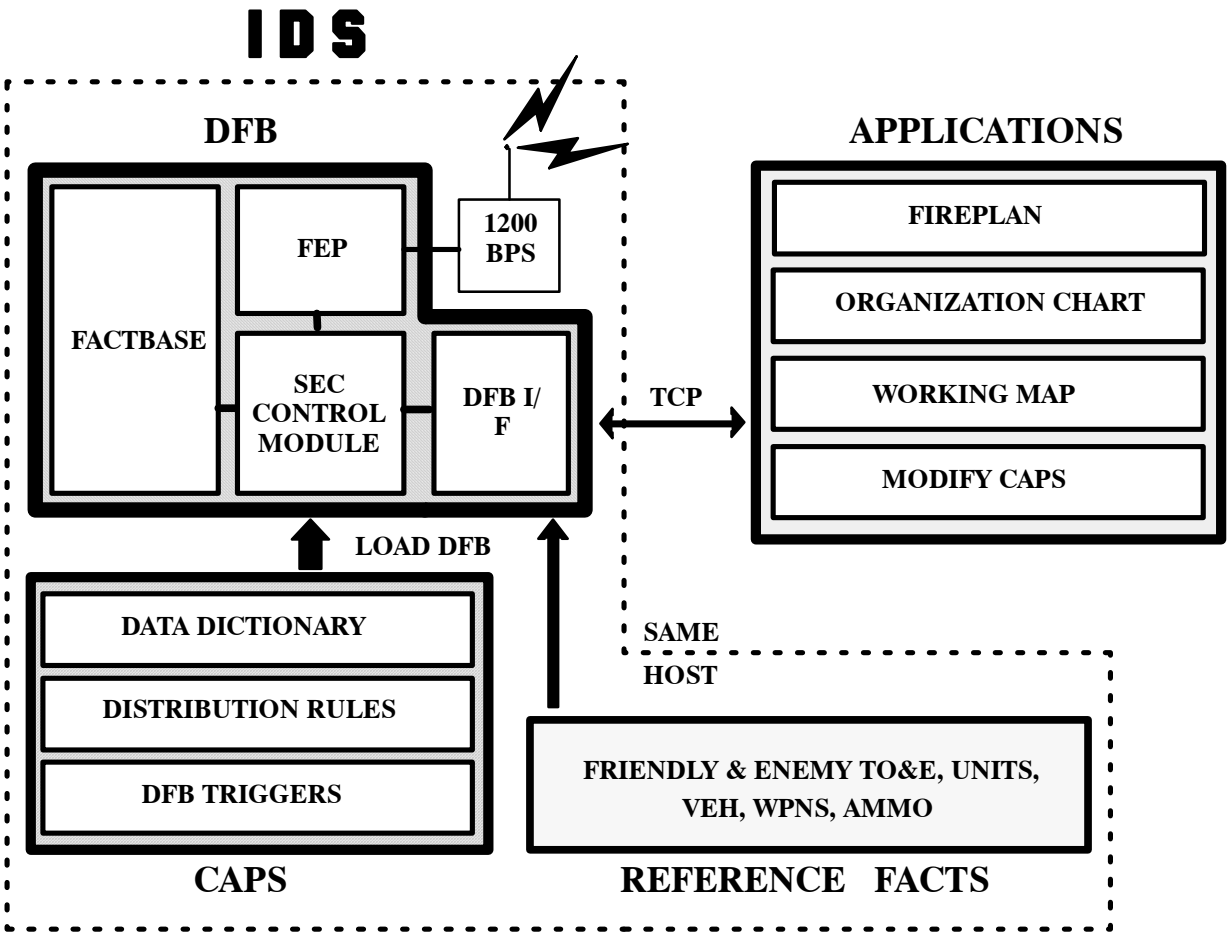

FIGURE 1. Block Diagram of IDS Software Components

Field Artillery" or the "USS Abc"). These facts have a fixed fact id because new units cannot be created, but the information within the fact can be updated. Therefore, "unit" facts are a special kind of reference material because the information they *contain* is dynamic (e.g., location) although the fact ids of the units are static. *Meta–facts* are a special version of dynamic facts that refer to potential modifications of other facts. Since facts are suppose to represent the real world situation, some type of fact must be available to describe future modifications of these facts. In this implementation there are two types of meta–facts (thus far): *planning* facts (actually called "what_if" facts) and *order* facts. These facts refer to other existing facts and then list fact items (fields) and new values for those fields. "What_if" facts are used to exchange ideas about future plans and order facts are used to command changes to the status of facts (e.g., Unit A move to location B). Everything about a node (a military unit) is also stored as a fact so that it can be exchanged with other nodes. This includes rules for how to distribute and accept information and parameters that control the exchange of information. **Figure 2** shows the relationship between the fact types (data abstractions) implemented thus far.

An important feature of the DFB is *triggers*. Triggers can be entered into the DFB by the application programs. A trigger is a set of criteria that can be compared to the information within a fact. Each time a fact is entered into the factbase it is compared to each trigger in the current list of user defined triggers for a criteria match. If a match is found, the owner of the trigger (an application program) is notified and provided the fact id of the fact

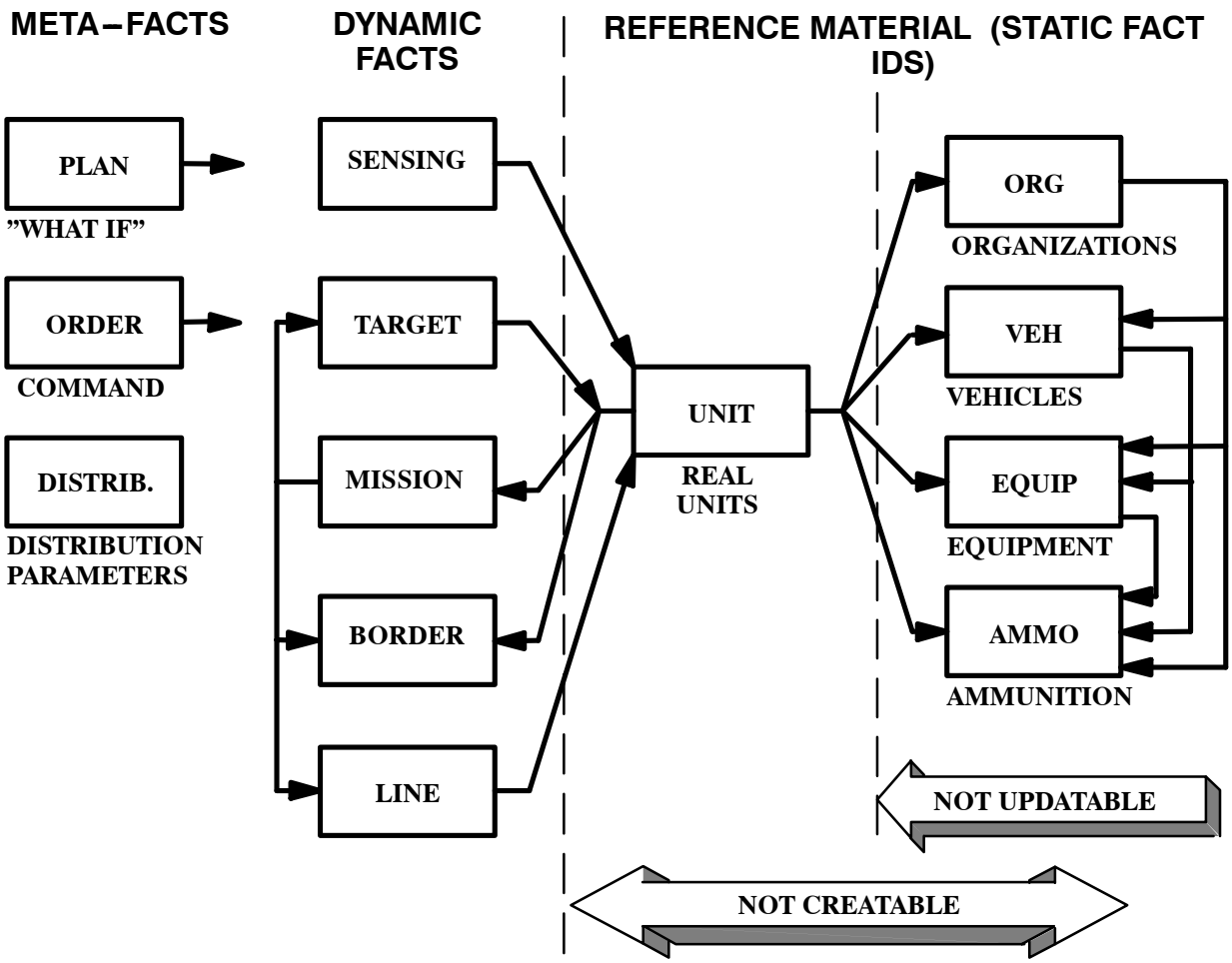

FIGURE 2: Data Abstractions (Fact Types) and Relationships

that set off the trigger. At that point, it is up to the application program to react (e.g., retrieve the fact from the factbase for further processing). A different form of triggers will also be used to by the Security Control Program to implement "over–hearing" (described below).

FACT EXCHANGE PROTOCOL

Information exchange between individual factbases is supported by the fact exchange protocol, or FEP. The FEP is a connectionless, reliable datagram protocol currently built on top of the standard DoD Internet Protocol (IP) and User Datagram Protocol (UDP) for convenience. The initial focus of the implementation is terseness, reliability, and overhearing. In the past, commanders and their staffs have kept themselves informed by simply listening to the voice transmissions occurring on several radio nets. Similarly, the collection of "free" digital data (at no cost in bandwidth) is a feature of the FEP. This allows pertinent overheard information to be collected thus preventing some information from being needlessly requested and retransmitted. A small "hack" will be made to IP to allow it to accept datagrams meant for other hosts. Those UDP datagrams destined for the well–known port associated with the DFB will then be passed on to the Security Control Module which will determine what overheard information is to be entered into the factbase; this will initially be implemented using triggers. A major obstacle to overcome is that overheard information must be meaningful. The IP protocol may arbitrarily fragment messages

and this causes a problem when trying to monitor information since partial facts can be meaningless. Obviously, no information transmission can be larger than the maximum transmission unit (MTU) of the channel over which it is to be sent. Since MTU values are easily determined in the low echelon tactical environment this does not present a problem as it might over a large internetwork. (In the BRL implementation, the channels will be Ethernet and FSK modems for VHF–FM radio, both with MTUs of approximately 1500 bytes.) Although unlikely, facts larger than the MTU will have to be broken at logical boundaries (i.e., by fact item) before being passed to the lower protocol layers so that overheard information will remain meaningful.

Due to the connectionless nature of the FEP, selective (out of sequence) acknowledgement of datagrams is provided since each datagram is a separate entity by itself. The implementation of a truly connectionless protocol has produced many interesting challenges. Concepts such as "windowing" require a new perspective, especially when dealing with communication channels with widely varying bandwidths and characteristics (e.g., 10 Mbps Local Area Net (LAN) versus 1200 bps FSK modems over the VHF–FM tactical radios used at the fighting echelons of the Army). For example, under the current strategy the window and wait time–out parameters for a datagram depends on both the hosts and the channel to which it was sent. This is because the regulating parameter may be the host if a high–speed link is used (e.g., a LAN) or the channel if a low speed link is used (e.g., VHF–FM radio). Obtaining these parameters (also stored as facts) is not a problem for this particular application although it would be for a large, heterogeneous internet.

A scheme called *maximum datagram packing* is being implemented. This data link level process "packs" each datagram with as many pending messages as possible based on the current MTU size for the communications channel being used. This provides significant bandwidth saving since much of an FM radio digital transmission is the "preamble" that typically precedes each individual datagram (message). This technique is made possible due to the overhearing capability of the FEP since every transmission can be heard by every node. A received datagram is parsed into messages at the data link level and then passed up to the higher protocol layers.

Future plans include implementation of a radio silence mode of operation just as has been common in voice communications. In this mode, a node may enter into EMCON mode status and continue to be sent information even though it is not returning acknowledgements. Upon leaving EMCON mode, a bulk acknowledgement scheme will be developed to update only the most recent information; the transfer of this information can be by any appropriate media (e.g., radio, floppy disk and motorcycle, etc.).

Multicast (the ability to send the same message to several recipients with only one transmission) is an active area of current networking research. Since this capability is commonly used in military voice communications, it is of specific interest to this application, especially in relation to the EMCON and network monitoring schemes. However, due to the attention this capability is receiving in academia it may not be included in the near–term versions of the FEP.

SECURITY CONTROL PROGRAM

The implementation of the Security Control Module, SCM, will be included in future research, but the initial focus will be on exchanges over the FEP (information to/from other factbases, that is, from outside the local host). The SCM will initially have two major functions: one, to determine what information should be sent to remote sources, and two, to determine what information obtained from remote sources will be entered into the factbase.

A key tenet of this project is that information should rarely (if ever) have to be requested because each unit is responsible for keeping adjacent units informed based on a set of rules concerning fact distribution. These rules strictly regulate the exchange of information so that only "significant" information is transmitted. For example, it is normally not significant to report every round of ammunition that an artillery battery fires (although this is currently done). More appropriately, ammunition status should be reported when it reaches predesignated values or rates. This capability is being implemented

using the distribution rules and the concepts of reporting depth and value thresholds.

Reporting depth constrains *what* status information is exchanged. For example, a reporting depth of 2 is traditionally used which indicates that every unit should have information about units 2 echelons below it; e.g., brigades track companies, divisions track battalions, etc.). This means that each unit sends status information to its parent about its direct subordinates; e.g., battalions send their parent brigade information about their companies. Therefore, a brigade would receive current updates about its companies, and if a status report is desired about a battalion, a roll–up would be done on the companies of that battalion. The purview of a particular unit based on its assigned reporting level and any other special cases is called the *distribution envelope* of that unit. A unit's distribution envelope can be dynamically modified based on the commanders desires; however, in a low bandwidth tactical environment increasing it will always incur a cost of increased signature and less bandwidth available to other units. The distribution envelope concept will also be used to assist in controlling the collection of overheard information since, for security reasons, it is not wise to incorporate all overheard information (i.e., should a node be captured).

When information should be exchanged is controlled by *value thresholds*. Thresholds are defined for the fact items (fields) within the fact types. They may be absolute or relative, and may be based upon either the current value in the factbase or the last value transmitted (the latter being the default). For example, an absolute threshold value could be placed on a unit's ammunition count to identify when it is less than 100 rounds (or 20% of its basic load), or a relative threshold value could be placed on its location to identify when a move of greater than 200 meters was made. A distribution rule is invoked (i.e., information exchanged) only when these threshold values are exceeded (i.e., the information is determined to be "significant") thus controlling how often valuable bandwidth is used to send information. It is believed that the use of reporting depth and threshold values can constrain the flow of information to reasonable levels, and this is one of the tasks of the Security Control Module. Of course, the values that should be used for these parameters are an equally interesting area of study (e.g., should reporting depth be constant or vary up and down the chain of command?).

Information received *from* remote sources is either intended for the receiving unit or overheard. Information intended for a receiving unit may be due to the invocation of a distribution rule or a response to an information request initiated by the receiving unit. Initially, simple schemes will be employed to determine if the information is worthy and trusted, but sophisticated rules will be the major area of study in future work. Eventually, the SCM will also be suspicious of the information being exchanged with the application programs (locally) as well as with remote sources.

CAPABILITY PROFILES

When a node is initialized, the information described above must be "loaded" into the distributed factbase. The fire support control capability profiles, or "CAPs", explicitly contain this fire support control information that describe the "personality" and capabilities of the particular node. Like all other information, this information is stored as facts to facilitate the exchange of this information. The CAPs include the *data dictionary* that defines the fact types in the factbase, the *fact distribution rules* that dictates what, when, where, and how facts are sent, replicated, or retrieved (e.g., reporting level, thresholds, etc.), and system level *triggers* to alert the DFB sub–programs when certain combinations of facts are present in a DFB. The CAPs can be modified dynamically (even from remote locations) thus allowing all the items previously mentioned to be maintained as best appropriate for a particular situation. For example, CAPs could be developed for offensive versus defensive scenarios, special situations, or training and stored locally at each hosts. CAP modifications could also be included as part of the information commonly associated with an "operations order", or in critical situations, they can be modified from another host to reconfigure the tactical command chain due to losses; of course, this also has serious security ramifications. It is envisioned that the CAPs, which basically implement standard operating procedures (SOPs), would be developed by

doctrine and tactics experts (e.g., US Army Training and Doctrine Command schools) and perhaps modified by specialists in the upper echelons of a particular unit (e.g., corps, division, or brigade); they would not normally be modified by the soldier in the field.

APPLICATION PROGRAMS

Above the distributed factbase sits the application programs; they retrieve information (facts) using queries and triggers and enter information by stating new facts or updating existing facts. All information is exchanged, either locally or remotely, using the factbase; information exchange is not conducted in the normal "message" fashion. Applications do not "send" information to other applications or factbases; rather, they enter information into their local factbase which will distribute that information to other hosts only if a distribution rule exists about that information. Likewise, the only way application programs receive information from a local factbase is to query for it or have a trigger set to alert them of new incoming facts. This is a radical change from the typical way of "doing business", but it allows predefined distribution rules to strictly control information exchanges, and therefore, the use of precious bandwidth. To modify the exchange of information one must update the distribution rules that control data distribution, in effect, modifying the CAPs. However, if the users wants to send a free–text character string, this can be easily facilitated (although discouraged). Hopefully, combat developers (military scientists) will eventually have data abstractions for most tactical situations so that free–text messages are rarely, if ever, required.

One of the primary purposes of an application program is to serve as the interface between the user and the DFB, that is, to display facts to the users and assist them in entering facts. Since facts are the data abstractions of military concepts in their "purest" form, it is not difficult to "show" a fact once it has been *correctly* represented; BRL implementation experience has been that if a fact is difficult to show then it is not correctly represented.

The interface between the applications and the factbase is implemented using standard TCP/IP sockets (standard DoD protocols). This allows the applications to reside either on the same host as the factbase or on a separate processor connected over a reasonably reliable data link medium (e.g., a LAN). So although inter–factbase information exchange is geared towards unreliable, low–bandwidth links using the FEP, the application to factbase information exchange is designed to operate most efficiently over reliable, high speed links (e.g., a dispersed command post using a LAN). To facilitate this capability, a utility called the "package protocol" was developed to handle most of the work in setting up the factbase interface (TCP/IP socket) for application program developers, and it is available for any machine providing it has TCP/IP and a "C" programming language compiler resident on it.

The preparation, maintenance, and dissemination of the information associated with a fire support plan in a maneuver operations order (OPORD) is an excellent vehicle to demonstrate the system's capabilities in a dynamic, real–time environment. Three major applications program are being developed: one, *Organization Facts* that shows organization diagrams and TO&E roll–ups of friendly and enemy unit, organization, vehicle, equipment, and ammunition facts; two, *Working Map* that shows geographical information on a map concerning the locations of friendly and enemy unit, sensing, line (fire control measures), border, and target facts and special features such as unit range fans; and three, *Fire Plan* that uses Organization Facts (Org Facts) and Working Map to display and enter the information commonly associated with fire support plans such as unit missions, objectives, routes, the enemy situation, fire control measures, planned targets, and more. The first two applications have been completed while Fire Plan is in the early implementation phase. All the applications make extensive use of of the Sun Microsystem workstation's graphics capability for both display and input. **Figures 3** and **4**show examples of the Org Facts and Working Map displays. A "conversational graphics" capability (between factbases) is supported through the use of the "what_if" and "order" meta–fact types (as described above) to refer to existing facts for discussing potential and required changes.

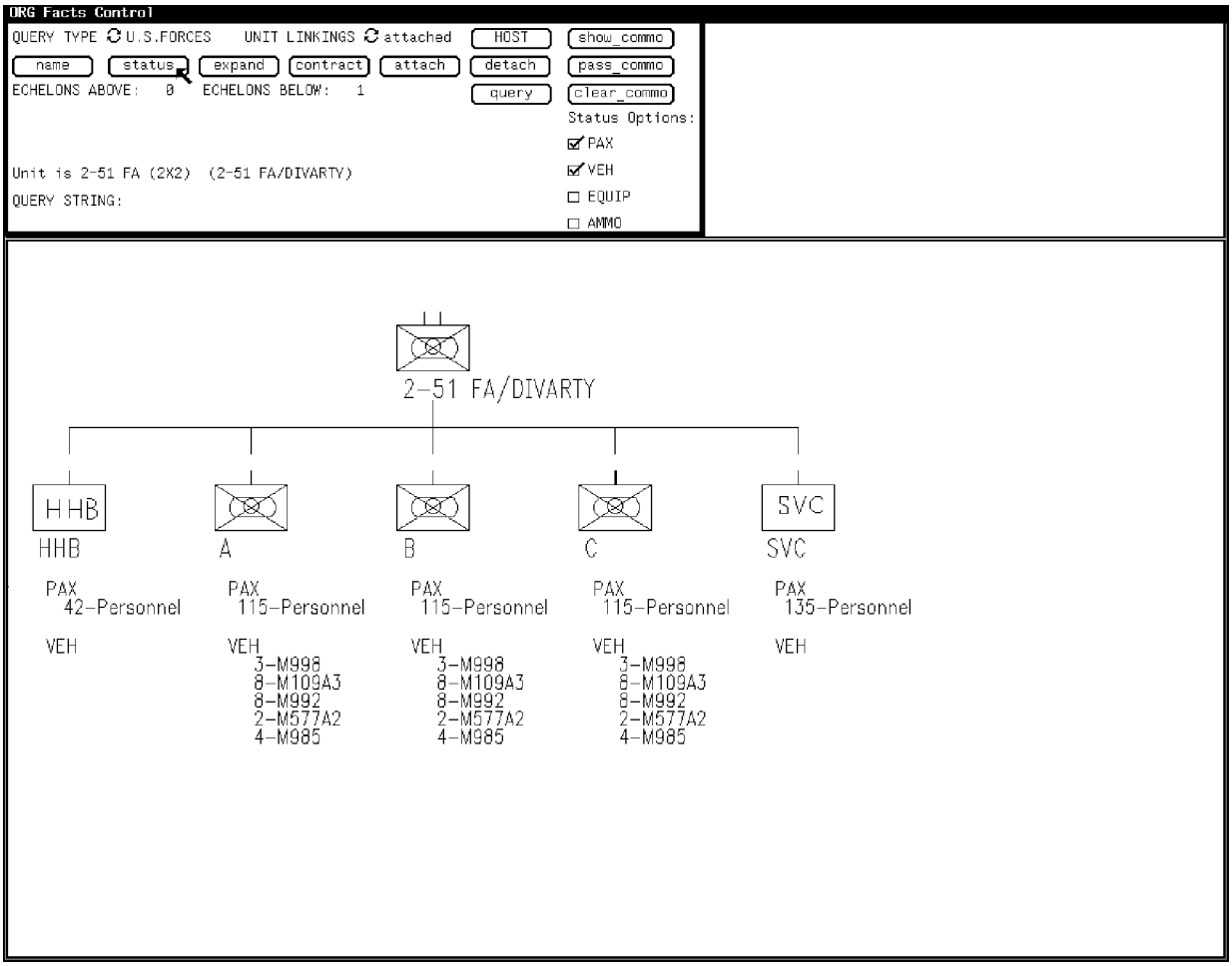

FIGURE 3: Sample Screendump from Org Facts Application

A fourth application is a "scenario driver" that will convert an input file of predefined events into facts and enter them into a separate factbase for distribution to the other nodes in the system. This will be used to load the five test nodes with friendly unit status and location updates, enemy sensings, targets, and a variety of other facts to provide the illusion of a mid–intensity battle. The tactical inputs were developed under contract and provide an unclassified, 60 minute, Fulda Gap battle between a friendly mechanized infantry brigade task force and an enemy tank division with one minute resolution down to the platoon level. From this master event list (i.e., "scenario"), events are selected to build a "vignette", that is, a particular blue perception of the battle; many vignettes can be built from the master list. During a test exercise, the traffic exchanged between the five nodes (five factbases) will be collected as well as data on the information exchanged between the application programs and the factbases. This data will be analyzed to study and evaluate the efficiency and worth of the factbase concept, the FEP features (e.g., overhearing), and the appropriateness of the data abstractions. In addition, subjective information will be obtained from the users concerning their ability to use the applications and to understand the situation that was presented. **Figure 5** illustrates a typical exercise configuration.

These application programs will be used together to demonstrate and evaluate the aforementioned DFB features and to assist the soldier by: identifying incoming information and alerting the operator, extracting current situation information

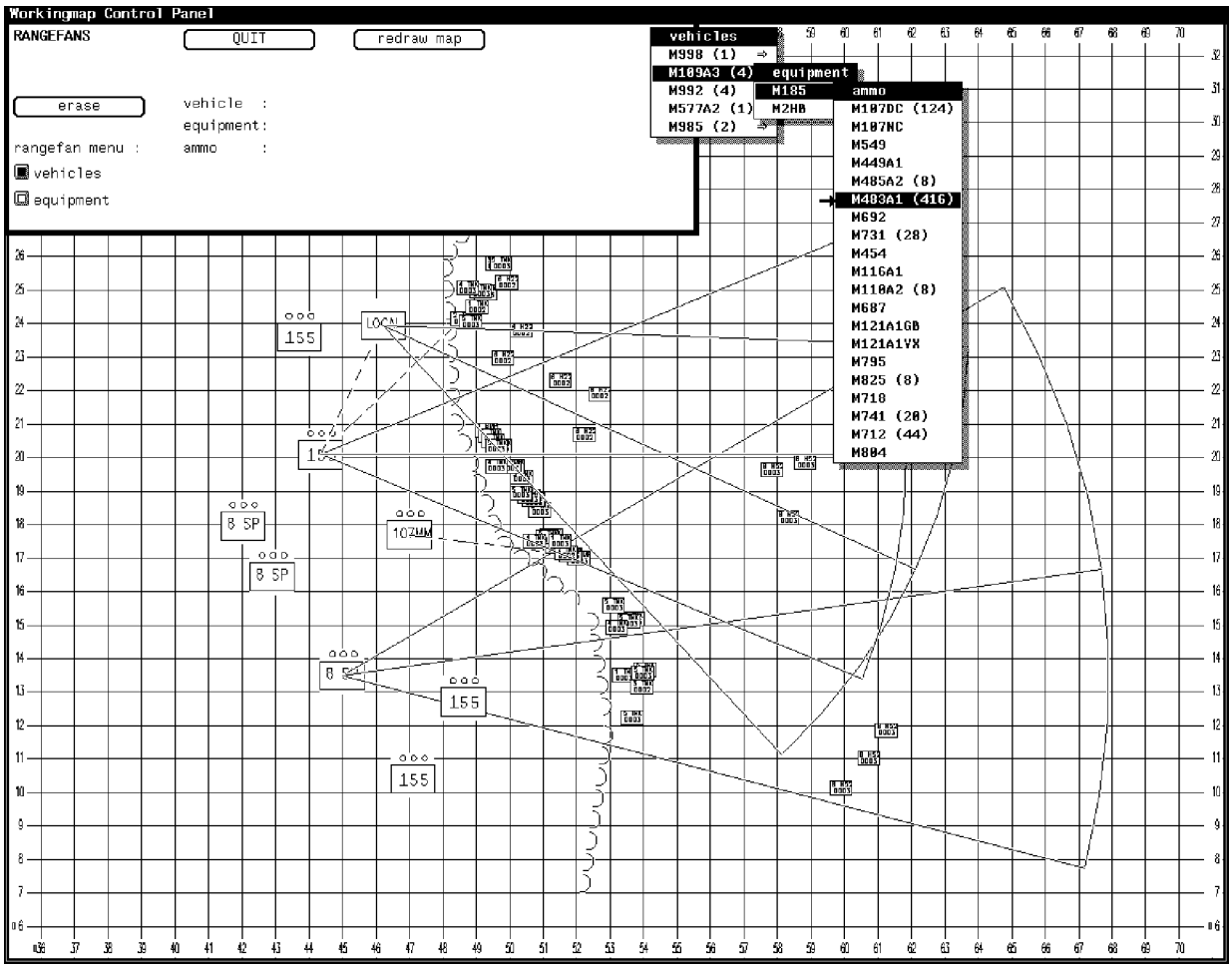

FIGURE 4: Sample Screendump of Working Map Application
(Map Background Turned Off)

from the factbase, graphically depicting unit mission and situational information, insuring that appropriate information is in the fact base, controlling the dissemination of fire support plan information, updating the prescribed CAPs, and making maximum use of graphics "tools" and other software available from the various ADCCP packages.

CONCLUSION

The goal of this project is to develop a experimental system that is responsive enough to the user that it will still be preferred during degraded modes of operation. At the fighting echelons of the brigade and below, information distribution is limited by the low bandwidth communications systems required for highly variable terrain (e.g., non–line–of–sight conditions). Hopefully, new information distribution technology concepts such as the distributed factbase, capability profiles, the features of the Fact Exchange Protocol (overhearing, and listening silence), and the application programs when combined with carefully developed data abstractions of military concepts will provide the capability to operate in spite of severely limited communications. If not, the equipment will be thrown aside during crisis situations and commanders and their staffs will continue to huddle in circles making figure drawings in the dirt when the battle begins.

ACKNOWLEDGEMENTS

This work is the result of the synergistic effect of the combined contributions of several members of

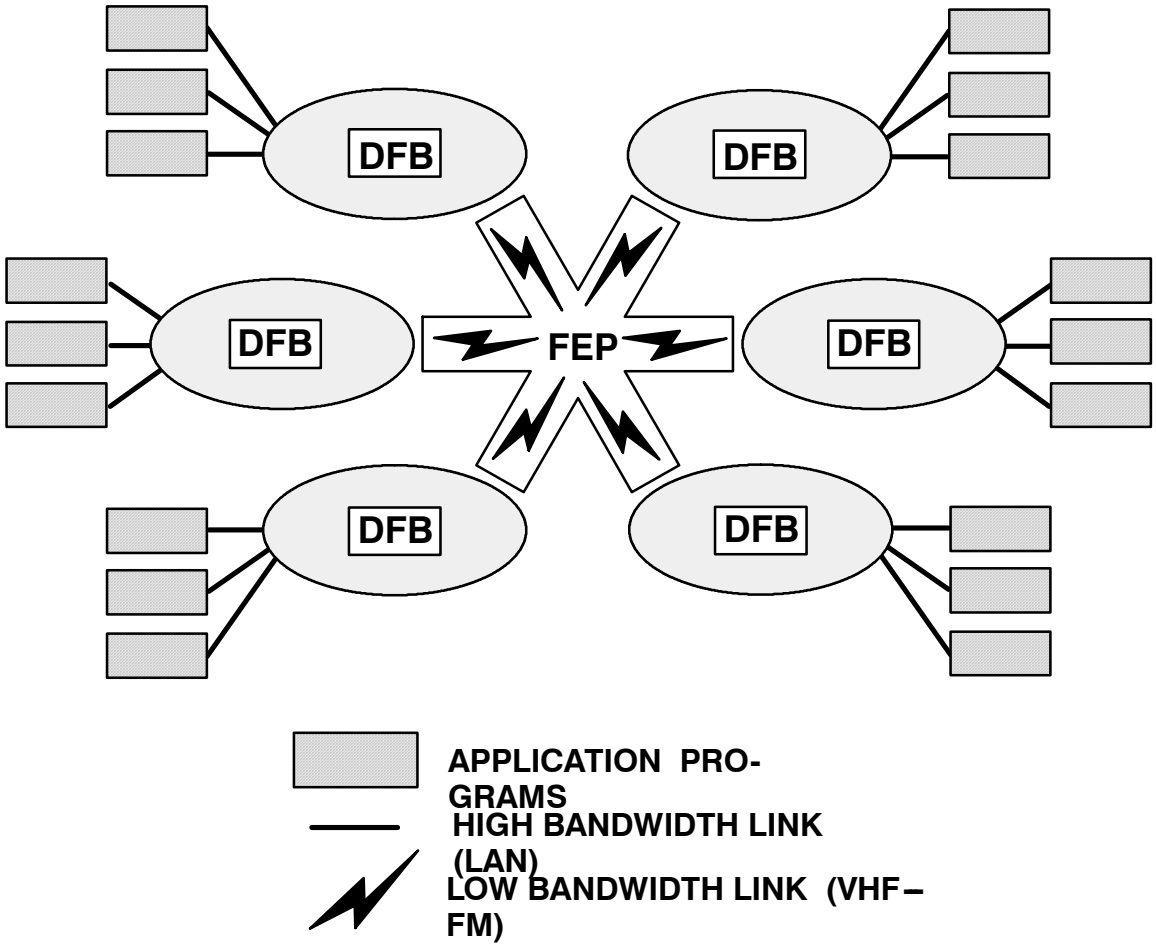

FIGURE 5. Typical Six Node DFB

the BRL's System Engineering and Concepts Analysis Division (past and present): Phil Dykstra, George Hartwig, Eric Heilman, Ginny Kaste, Don Merritt, Mike Muuss, Joe Pistritto, Ken Smith, Wendy Winner, and Mike Zoll.